KATHOLIEKE UNIVERSITEIT
LEUVEN

Faculty of Business and Economics

On the interaction between railway scheduling and resource flow networks

Wendi Tian and Erik Demeulemeester

DEPARTMENT OF DECISION SCIENCES AND INFORMATION MANAGEMENT (KBI)

KBI 1019

On the interaction between railway scheduling and resource flow networks

Wendi Tian[1, 2] and Erik Demeulemeester[1]

[1]*Katholieke Universiteit Leuven, Research Center for Operations Management, Leuven, Belgium*

[2]*School of Management, Huazhong University of Science and Technology, Wuhan, China*

wendi.tian@hotmail.com and Erik.Demeulemeester@econ.kuleuven.be

Abstract: In previous research (Tian & Demeulemeester, 2010), we have shown that in realistic situations railway scheduling improves both the stability and the expected project length over roadrunner scheduling. In this paper, we introduce the concept of resource flow networks in this analysis and determine what the impact is of the resulting combinations on the average project length, the standard deviation of the project length, the timely project completion probability and the stability cost. Extensive computational results will be presented on both small and larger projects and statistic analysis will be conducted by using SAS PROC GLM.

Keywords*: railway scheduling, roadrunner scheduling, resource flow networks, DTRTP*

1 Introduction

In the past, quite a lot of research has been performed on the resource-constrained project scheduling problem (RCPSP, referred to as $m,1|cpm|C_{max}$ in the classification scheme of Herroelen et al. (1999, 2000)). In this research, the duration of each activity as well as its resource requirements are usually assumed to be known in advance. In practice, however, it often occurs that activities can be executed with different amounts of resources, resulting in different possible durations (e.g. research and development projects, new construction projects). In these cases, activities are assumed to be given a specific work content instead of both a deterministic duration and deterministic resource requirements, whereby different combinations of durations (e.g. days) and resource requirements (e.g. units/day) could be specified, as long as the product of the duration and the corresponding resource requirement at least equals the activity´s work content. This kind of problem with only one single renewable resource type was introduced by De Reyck (1998), De Reyck et al. (1998) and Demeulemeester et al. (2000), and is known as the discrete time/resource trade-off problem (DTRTP, referred to as $1,1|cpm,disc,mu|C_{max}$ in the classification scheme of Herroelen et al. (1999, 2000)). For reviews of the DTRTP, we refer to Demeulemeester et al. (2000), De Reyck et al. (1998), Ranjbar and Kianfar (2007), Long and Ohsato (2008) and Ranjbar et al. (2009).

However, typically these research efforts focus on developing algorithms for finding a good or optimal schedule from the millions of possible mode combinations in this DTRTP. In addition, there might be many different schedules with the same optimal project length in the

deterministic version of the problem, which might result in different average project lengths when executing the obtained schedules in a stochastic environment. To the best of our knowledge, there is no research on scheduling policies in a stochastic DTRTP environment except for the paper by Tian and Demeulemeester (2010). In this paper, it has been shown that in a stochastic DTRTP environment with condense schedules railway scheduling improves both the stability and the expected project length over roadrunner scheduling considering all optimal baseline schedules with the same project length.

Roadrunner scheduling is also named relay racer behavior or roadrunner mentality and is typically applied during the execution of a Critical Chain/Buffer Management (CC/BM) schedule (Goldratt, 1997). In this approach, the activities of the project should start as soon as possible when all their predecessors have finished and when enough resource units are available to start. Railway scheduling is a quite different strategy that is derived from train schedules. Trains are scheduled to leave a railway station at a certain time and are not allowed to do so at an earlier time. So railway scheduling implies that every activity of the project should not start earlier than its planned starting time. The roadrunner scheduling policy is typically executed in the hope of decreasing the expected project length, while the railway scheduling policy is mainly introduced in order to increase the stability of the project. In Tian and Demeulemeester (2010), we have enumerated all mode combinations in order to obtain those that result in a minimal project length in the deterministic case and we have applied the concept of CC/BM to the DTRTP. The computational results indicated that the mode combinations and the execution policies we chose had a huge impact on the final results although they originated from baseline schedules with the same optimal project length. Additionally, it was shown that in this project environment railway scheduling should typically be combined with relatively small feeding buffers (0% to 10%). Moreover, we might think that the roadrunner mentality is executed in the hope of decreasing the expected project length. However, the computational results indicate clearly that for the very condense schedules that resulted for these problems (and that project managers might prefer in practice) railway scheduling performs better than roadrunner scheduling, not only for the stability cost and the standard deviation of the project length (as could be expected), but also for the average project length and the timely project completion probability. The main reason for this quite unexpected behaviour is that, compared to roadrunner scheduling, railway scheduling can avoid more that some activities jump over other activities, resulting in perturbations of the baseline schedule which result in longer project lengths as the same condense schedules could not be continued.

This paper continues our research on the scheduling policies in a stochastic DTRTP environment. Another way to avoid that activities jump over each other during the execution of a project is to define resource flow networks for the baseline schedule (for more information on resource flow networks we refer to Artigues et al. (2003), Leus (2003), Leus and Herroelen (2004), Policella et al. (2004), Policella (2005) and Deblaere et al. (2007)). Resource flow networks, which determine how renewable resource units are passed along from one activity to another in the baseline schedule, indeed determine the sequence of allocations of resource units over time. Executing a project according to a scheduling policy that is based on a resource flow network is therefore quite different from using a scheduling policy in combination with a priority list. Indeed, in the first case, an activity can start as soon as all activities that should pass resource units along to it have finished. In the second case, every time an activity finishes, it is determined which of the remaining activities (in the order of the priority list) can be started as all its predecessors have completed and enough resource units are available. The aim of this paper is to analyze what the impact is of applying resource flow networks in combination with roadrunner and railway scheduling on the average project length, the standard deviation of the project length, the timely project completion probability and the stability cost.

The remainder of this paper is organized as follows. The next section gives a literature review about resource flow networks and some related topics. The basic DTRTP, the mode selection and the optimal baseline schedules are described in section 3. Section 4 introduces the concept of resource flow networks as well as two resource flow network algorithms that will be used in this paper. The set-up of our computational experiments is described in section 5, while the results of the computational experiments are shown in section 6. Section 7 analyzes what the impact is of the resulting combinations on the average project length, the standard deviation of the project length, the timely project completion probability and the stability cost. The last section provides our overall conclusions and offers some suggestions for further research.

2 Literature review

In this paper, we introduce the concept of the resource flow networks to the stochastic DTRTP environment and try to analyze the railway scheduling and roadrunner scheduling more in-depth. As the DTRTP considers only one single renewable resource type with multiple execution modes, it is a subproblem of the MRCPSP (multi-mode RCPSP, referred to as $m, 1T|cpm, disc, mu|C_{max}$ in the classification scheme of Herroelen et al. (1999,

2000)). There are many algorithms for the general MRCPSP that can also be used for solving the DTRTP. Chapter 8 of the project scheduling research handbook of Demeulemeester and Herroelen (2002), Brucker et al. (1999), Hartmann and Briskorn (2010) and Van Peteghem and Vanhoucke (2010) give a literature review of procedures for the MRCPSP. A literature review on the DTRTP (solution approaches for the DTRTP and algorithms for solving the MRCPSP) and the work content can be found in Tian and Demeulemeester (2010).

Resource flow networks, which indicate how renewable resource units are being passed along between the various project activities in the baseline schedule, have been proven to be a good method for the resource allocation of the RCPSP. However, the literature on resource flow networks is relatively sparse. Artigues et al. (2003) presented a simple method to generate a feasible resource flow network without considering any measure of performance by extending a parallel schedule generation scheme to derive the flows during scheduling. A feasible resource flow network was obtained very easily by rerouting flow quantities iteratively from the baseline schedule generation, which is the reason why we choose this algorithm in our computational experiment. A branch-and-bound algorithm for the resource allocation which just considers a single resource type was proposed by Leus (2003) and Leus and Herroelen (2004), aiming to minimize the stability cost for the project with variable activity durations. Policella (2005) (see also Policella et al. (2004)) proposed a procedure referred to as chaining for constructing a chained partial order schedule (POS) from a given precedence and resource feasible baseline schedule. Deblaere et al. (2007) developed three integer programming based heuristics (MinEA, MaxPF and MinED) and one constructive resource allocation procedure (MABO, myopic activity-based optimization) to protect a given baseline schedule against activity duration variability. These heuristics have different objective functions: the MinEA heuristic aims to minimize the number of direct extra precedence relations imposed by the resource allocation decision, the MaxPF heuristic is trying to maximize the sum of the pairwise floats between the activities, the MinED heuristic intends to minimize an approximation of the weighted expected activity starting time deviations, and the MABO heuristic tries to construct a resource flow network with the objective of minimizing the project execution costs. In addition, the MABO heuristic was demonstrated to be a good procedure for the stability cost objective within a short computation time, which is the main reason for the choice of including this algorithm in our computational experiments. Our overview of the literature on the resource flow networks reveals that research efforts focus on the algorithm of generating resource flows in the RCPSP. In this paper, we try to apply the concept of resource flow networks to the DTRTP.

3 Problem description and baseline schedule

In this section, the discrete time/resource trade-off problem (DTRTP) will be described formally as well as the way in which we select the mode combinations and the optimal baseline schedules that will be used in the section on computational results.

3.1 Basic DTRTP

Given a specified work content W_i for activity i, all M_i efficient execution modes for its execution are determined by time/resource trade-offs. Activity *i,* when performed in mode m $(1 \leq m \leq M_i)$, has a duration d_{im} and requires a constant amount r_{im} of the renewable resource during each period it is in progress, such that $r_{im} \times d_{im}$ is at least equal to and as close as possible to W_i $(r_{im} \times d_{im} \geq W_i)$. A mode is called efficient if every other mode has either a higher duration or a higher resource requirement. It is assumed that the dummy start and dummy end activity have only one execution mode with zero duration and zero resource requirement. There is only one single renewable resource type that has a constant per period availability a. The problem is to find a deterministic schedule with suitable durations and corresponding resource requirements under the precedence and resource constraints with the objective of minimizing the project length. Using the classification scheme of Herroelen et al. (1999, 2000), the problem is denoted as 1,1|*cpm,disc,mu*|C_{max}. The DTRTP has been shown to be strongly NP-hard (De et al., 1997).

We assume that projects are represented in activity-on-the-node representation, where the precedence constraints are of the finish-start type with a zero time-lag. An example network with ten real activities is given in Fig. 1: nodes 0 and 11 are the dummy start and end activities, respectively. There is only one renewable resource type with 10 available units in each time period. The number above the node represents the mean work content which is obtained randomly in a way that will be described later on in section 5.

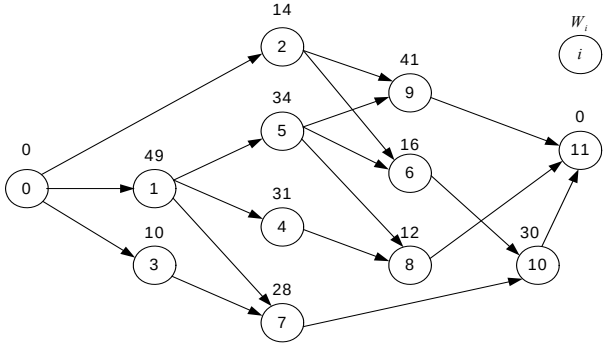

Fig. 1 An example network

3.2 Mode selection and optimal baseline schedule

There are millions of combinations of efficient modes for the DTRTP and there are also a lot of exact and heuristic procedures available for obtaining good schedules for this problem. In this paper, we continue our previous research (Tian & Demeulemeester, 2010) to analyze railway scheduling and roadrunner scheduling. So, we use the same way (a slightly adapted version of the branch-and-bound procedure that was developed by Demeulemeester et al. (2000) for the *DTRTP*) to select the mode combinations and obtain all optimal baseline schedules. Table 1 gives all optimal mode combinations for the example in Fig. 1 and the corresponding optimal early start baseline schedules are shown in Table 2.

Table 1: All optimal mode combinations for the example in Fig. 1

Choice	Modes chosen	Makespan
1	<0,0>,<5,10>,<7,2>,<5,2>,<8,4>,<9,4>,<4,4>,<7,4>,<3,4>,<7,6>,<3,10>,<0,0>	27
2	<0,0>,<5,10>,<2,7>,<1,10>,<11,3>,<5,7>,<4,4>,<4,7>,<3,4>,<7,6>,<3,10>,<0,0>	27
3	<0,0>,<5,10>,<7,2>,<1,10>,<11,3>,<7,5>,<4,4>,<4,7>,<3,4>,<7,6>,<3,10>,<0,0>	27
4	<0,0>,<5,10>,<7,2>,<5,2>,<4,8>,<6,6>,<8,2>,<14,2>,<2,6>,<7,6>,<3,10>,<0,0>	27
5	<0,0>,<5,10>,<5,3>,<10,1>,<8,4>,<5,7>,<8,2>,<4,7>,<2,6>,<14,3>,<3,10>,<0,0>	27
6	<0,0>,<5,10>,<2,7>,<1,10>,<11,3>,<5,7>,<4,4>,<4,7>,<6,2>,<14,3>,<6,5>,<0,0>	27
7	<0,0>,<5,10>,<7,2>,<1,10>,<11,3>,<7,5>,<4,4>,<4,7>,<6,2>,<14,3>,<6,5>,<0,0>	27

Table 2: The optimal starting times for the mode combinations of table 1

Choice	act 0	act 1	act 2	act 3	act 4	act 5	act 6	act 7	act 8	act 9	act 10	act 11
1	0	0	10	5	5	5	20	13	14	17	24	27
2	0	0	6	5	6	8	20	13	17	17	24	27
3	0	0	10	5	6	10	20	6	17	17	24	27
4	0	0	9	5	5	9	16	10	15	17	24	27
5	0	0	5	10	10	5	10	20	18	10	24	27
6	0	0	6	5	6	8	13	17	21	13	21	27
7	0	0	6	5	6	6	13	17	21	13	21	27

4 Resource flow networks

In this section, we will first introduce the concept of resource flow networks more formally. Later on, two algorithms (Artigues et al.´s algorithm and the MABO algorithm), which are chosen from the list of existing resource flow network algorithms, are described and applied on the example problem.

4.1 Resource flow networks

A resource flow network describes how the renewable resources are transferred from one activity to another in the baseline schedule. It has the same set of nodes as the original project network. Besides the original precedence arcs, there are some resource arcs which start from node *i* to node *j* if there is a resource flow f_{ijk} of any resource type *k* from activity *i* to activity *j*. For the DTRTP, there is only one single resource type, so it is not necessary to consider the resource type *k*. We assume that the sum of all flows out of the dummy start activity equals the sum of all flows into the dummy end activity, both of them being equal to the resource availability *a* and the formulation is as follows: $\sum_{i\in N} f_{0i} = \sum_{i\in N} f_{in} = a$. For every non-dummy activity *i*, the sum of flows into this activity must equal the sum of flows out of this activity, which must equal the resource requirement of this activity r_{im}. The resource flow constraint is: $\sum_{j\in N} f_{ji} = \sum_{j\in N} f_{ij} = r_{im,}\ \forall i \in N\backslash\{0, n\}$.

Fig. 2 shows a feasible resource flow network for the sixth choice of mode combinations of the example schedule in Fig. 1. Fig. 2(a) is the network representation, while Fig. 2(b) is the resource profile representation. In Fig. 2(a), we can see that the solid arcs represent the original precedence relations, while the dashed arcs represent the extra precedence relations which are obtained from the resource flow constraints. The number on the arcs *(i, j)* are the amounts of the resource flows from activity *i* to activity *j*, so we get the following non-zero flows: $f_{0,1}=10, f_{1,3}=10, f_{3,2}=7, f_{3,4}=3, f_{2,5}=7, f_{5,6}=4, f_{5,9}=3, f_{4,7}=3, f_{6,7}=4, f_{7,8}=2, f_{7,10}=5, f_{8,11}=2, f_{9,11}=3, f_{10,11}=5$. Fig. 2(b) contains the same information as Fig. 2(a), but in a different format. For this schedule, as the resource profile of the schedule is very condense, there is only one feasible resource flow network. For other schedules, there might be many different resource flow networks as will be shown in section 4.2.

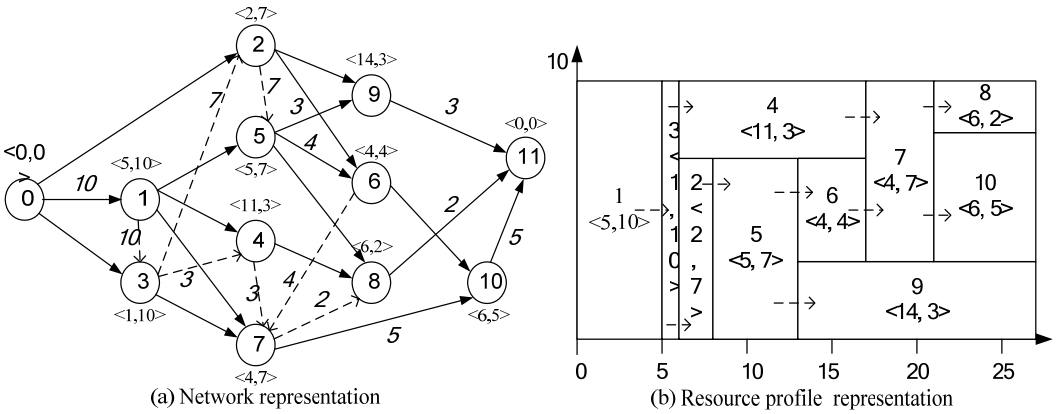

Fig. 2 A feasible resource flow network based on the schedule of the sixth choice of mode combinations

4.2 Resource flow networks algorithms

In the literature review, we observed that there are different heuristics for determining the resource flow networks. However, here we decide to use Artigues et al.´s resource flows algorithm and Deblaere et al.´s MABO algorithm to obtain the resource flow networks in our computational experiment. The main reasons to choose them have already been mentioned in the literature review and are as follows: Artigues et al.´s algorithm is able to generate a feasible resource flow in a very simple way, while the MABO algorithm is a good heuristic which tries to construct a robust resource flow network by considering one activity at a time, each time determining the best way to transfer resource units from previously finished activities. The MABO algorithm has already been proven to be a good procedure for optimizing the stability cost by Deblaere et al. (2007). In the description of both algorithms we will remove all references to the resource type k as the DTRTP only considers one resource type.

4.2.1 Artigues et al.´s algorithm

A simple method to generate a feasible resource flow by extending the parallel generation scheme was proposed by Artigues et al. (2003). This initial resource flow can be easily obtained from the initial baseline schedule generation. The resource flow f_{0n} is initialized with the resource availability a, while all the other flows are set to be 0. They define δ as the set of time instances in the input schedule that correspond with activity start times: $t \in \delta \; if \; \exists j \in N: t = s_j$. The algorithm of a feasible resource flow generation is given in table 3.

Table 3: Artigues et al.´s resource flow algorithm

Algorithm: Artigues et al.´s resource flow algorithm

for increasing t in δ **do**

  for $j = 1$ to $(n - 1)$ **do**

    if $(s_j == t)$ **then**

      $req = r_{jm}$;

      $i = 0$;

      while $(req > 0)$ **do**

        if $(s_i + d_{im} \leq s_j)$ **then**

          $q = min(req; flow_{in})$;

          $req = req - q$;

          $flow_{in} = flow_{in} - q$;

          $flow_{ij} = flow_{ij} + q$;

          $flow_{in} = flow_{in} + q$;

$i = i + 1$;

As there exists only one feasible resource flow network for the sixth choice of mode combinations which can be seen in Fig. 2, we choose another baseline scheme (the fifth choice of mode combinations) to generate resource flow networks in order to show that different resource flow networks can be obtained by using different algorithms.

Fig. 3 represents the resource flow network that is obtained by using Artigues et al.´s resource flow algorithm for the fifth choice of mode combinations of the example in Fig. 1. From the baseline schedule scheme, we can easily obtain non-zero flows by using Artigues et al.´s algorithm: $f_{0,1}=10$, $f_{1,2}=3$, $f_{1,5}=7$, $f_{2,4}=3$, $f_{5,3}=1$, $f_{5,4}=1$, $f_{5,6}=2$, $f_{5,9}=3$, $f_{4,8}=4$, $f_{6,8}=2$, $f_{3,7}=1$, $f_{8,7}=6$, $f_{7,10}=7$, $f_{9,10}=3$, $f_{10,11}=10$. Fig. 3(a) and Fig. 3(b) represent the resource flow network in two different ways.

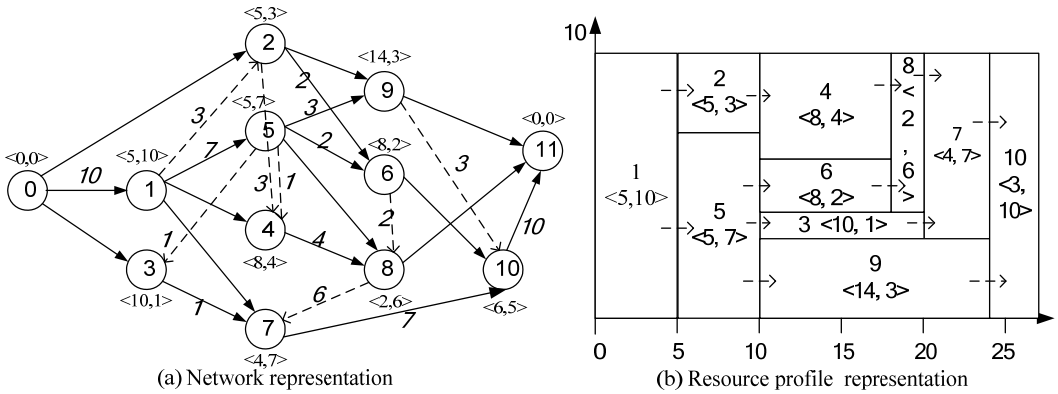

Fig. 3 The resource flow network obtain from Artigues et al.´s algorithm for the fifth choice of mode combinations

4.2.2 MABO algorithm

The MABO (myopic activity-base optimization) heuristic was proposed by Deblaere et al. (2007). This procedure which just considers the locally best possible resource allocation for the activity doesn't take the other activities into account. In comparison with other existing resource allocation procedures, it works activity-based rather than resource-based. MABO consists of three steps to be executed for each activity. Step1 examines whether the current predecessors of the activity may release sufficient resource units to satisfy the resource requirements of the activity. Step2 is trying to find additional resource units if there are no sufficient resource units assigned in step1, resulting in new precedence constraints (resource arcs) which have to be added. And the last step is to define the resource flows f_{ij} from predecessor activity i to activity j. The detailed steps of the MABO algorithm can be seen in Table 4. Some notation that will be used in the algorithm is elaborated here. A_R represents

the set of resource arcs, while A_U represents the set of unavoidable arcs. The number of resource units $alloc_0$ that may be transferred from the dummy start activity is initialized to the resource availability a.

Table 4: MABO resource flow network algorithm

Algorithm: MABO

Initialize: $A_R = A_U$ and $alloc_0 = a$

For each activity $i \in N \setminus \{0, N\}$, calculate the estimated stability cost contribution ec_i

Sort the project activities by increasing s_j (tie-break: decreasing c_j)

For every activity j in the sorted list

1. Calculate $Avail_j(A \cup A_R) = \sum_{\forall i:(i,j)\in A\cup A_R} alloc_i$
2. If $Avail_j(A \cup A_R) < r_{jm}$

    2.1 Define the set of arcs H_j with $(h,j) \in H_j \Leftrightarrow$

    $(h,j) \notin A \cup A_R$

    $s_h + d_{hm} \leq s_j$

    $alloc_h > 0$

    2.2 Determine all minimal subsets $H_j^1, H_j^2, \dots, H_j^q \subseteq H_j$ such that

    $Avail_j\left(A \cup A_R \cup H_j^i\right) \geq r_{jm},\ i = 1, \dots, q$

    2.3 Identify the subset $H_j^* \in \{H_j^1, H_j^2, \dots, H_j^q\}$ such that

    $Stability\_cost(A \cup A_R \cup H_j^*)$ is minimized

    2.4 Add H_j^* to A_R

3. Allocate resource flows f_{ij} to the arcs $(i,j) \in (A \cup A_R)$:

    3.1 Sort predecessors i of j by:

    Increasing number of successors l of i with $s_l > s_j$ and $r_{lm} > 0$

    Tie-break 1: Decreasing finish times $s_i + d_{im}$

    Tie-break 2: Decreasing variance σ_i^2 of d_{im}

    Exception: Activity 0 is always put last in the list

    3.2 While $alloc_j < r_{jm}$

    Take next activity i from the list

    $f_{ij} = \min(alloc_i, r_{jm} - alloc_j)$

    Add f_{ij} to $alloc_j$

    Subtract f_{ij} from $alloc_i$

From the baseline schedule scheme of the fifth choice of mode combinations, we can easily obtain some non-zero unavoidable flows: $f_{0,1}=10$, $f_{1,2}=3$, $f_{1,5}=7$, $f_{5,4}=1$, $f_{4,8}=4$, $f_{6,8}=2$, $f_{3,7}=1$,

$f_{8,7}=7, f_{9,10}=3, f_{7,10}=7, f_{10,11}=10$. These arcs {(0,1), (1,2), (1,5), (5,4), (4,8), (6,8), (3,7), (9,10), (8,7), (7,10), (10,11)} were called Unavoidable Arcs by Deblaere et al. (2007).

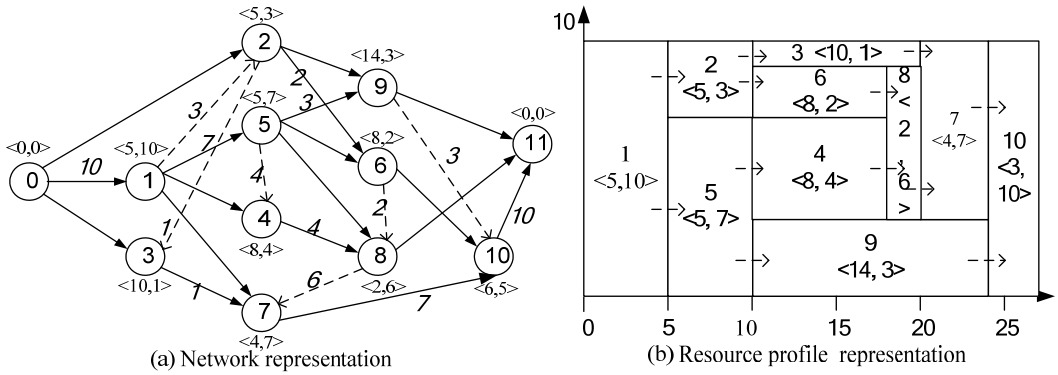

Fig. 4 The resource flow network obtained from the MABO algorithm for the fifth choice of mode combinations

In this example, the only thing left is to determine how the resource flows transfer from the parallel activities 2 and 5 to the parallel activities 3, 4, 6 and 9. There are many different ways to transfer the flows. Take activity 3 for example, there are two eligible activities 2 and 5 which can transfer a resource flow of one unit to activity 3. Simulations show that subsets {(2, 3)} and {(5, 3)} have the same instability cost, so the MABO procedure arbitrarily chooses the first subset {(2, 3)}. The network and resource profile representation of the resource flow network generated by the MABO algorithm for the fifth choice of mode combinations is shown in Fig. 4. Clearly, we can see that this resource flow network is different from the resource flow network that is obtained by Artigues et al.´s algorithm and that was represented in Fig. 3.

5 Computational set-up

In Tian & Demeulemeester (2010), the computational results of the first experiment indicated that railway scheduling should typically be combined with relatively small feeding buffers (0% to 10%) while the second computational experiment indicated that a choice for the starting time priority list and the first critical chain priority list resulted in the best outcomes when considering all four performance indicators. We decide to retain the starting time priority list and the first critical chain priority list with a feeding buffer of 10% combined with roadrunner scheduling and railway scheduling, and to compare these with a resource flow networks based approach in order to see what the impact is of the resulting combinations on the performance. In this paper, we continue to use four performance indicators to measure the

project outcomes, namely the average project length (APL), the standard deviation of the project length (SDPL), the timely project completion probability (TPCP) and the stability cost (SC). The average project length is the average completion time of finishing the whole project while satisfying all resource and precedence constraints, while the timely project completion probability is the probability of finishing the project within the due date (chosen to be 20% above the optimal project length of the deterministic case), which is an indicator of the efficiency or effectiveness of the project. The standard deviation of the project length is a measure of the variability of the project length, while the stability cost is the weighted sum of the penalty costs of the deviations between the realized starting times and the planned starting times. For these four performance indicators, the smaller the average project length, the standard deviation of the project length and the stability cost, and the larger the timely project completion probability, the better the outcome of the project. The function of each performance indicator is shown in Table 5. Please remark that the weighting function for the stability cost comes from Van de Vonder et al. (2008). It belongs to a triangular distribution with $P(w_i = q) = (21 - 2q)\%,\ \ q \in \{1,2,\cdots,10\}$, where w_i is the penalty weight. The distribution results in a higher probability for low weights and in an average weight $w_{ave} = 3.85$. The weight of the dummy end activity denotes the marginal cost of not completing the project within the due date and is set to 38. There will be no extra stability cost if finishing the project early or at the due date.

As test sets for assessing the execution policies described in this paper, we use the well-known PSPLIB set of project network instances (Kolisch & Sprecher, 1996). Because of the quite large computation times, we randomly chose 100 instances from the set of MRCPSP instances with 12 activities, while another 100 instances with 32 activities were selected randomly from the RCPSP instances. The resource availabilities of 10, 15 and 20 are considered in our computational experiment. The mean work content and the standard deviation of the work content for each instance were randomly generated from the uniform distribution between 10 and 50 and 1 and 5, respectively. When performing the simulation runs for each schedule, we assume that the work content of every activity *i* belongs to a normal distribution with mean μ_i and standard deviation σ_i. If σ_i is very large and μ_i is fairly small, a negative work content might be generated. In the real world, this is impossible and meaningless. So, during the simulation, whenever the obtained work content from this normal distribution turned out to be negative, it is set to zero and thus the activity does not have to be executed (this happened 1196 times in total in our experiments, namely 360 for the

small instances and 836 times for the larger instances: for both sets 100 random networks were simulated a 1000 times with 10 or 30 real activities, meaning that the occurrence of a zero duration for a single activity was 0.036% and 0.028% respectively).

For the test set with 12 activities, we can obtain all optimal baseline schedules by using the adapted branch-and-bound procedure that was already mentioned in section 3.2. However, for the test set with 32 activities, as the number of mode combinations of the DTRTP is $O(|M|^n)$, where $|M|$ denotes the maximum number of efficient modes that can be assigned to each activity and n denotes the number of real activities in the project, there are so many efficient modes that it is extremely difficult to obtain all optimal solutions by branch-and-bound procedure within a reasonable computation time. Based on this argument, for these 32 activity problems we obtain some feasible good solutions by the Tabu Search procedure of De Reyck et al. (1998) with a maximum allowable computation time of 30 minutes. Remark that these feasible good schedules are also very condense. The parameter settings used for the experiments are summarized in Table 5.

Table 5: Parameter settings for the experiments

Control parameter	Values	Control parameter	Values or functions
No. of activities (include the dummy activities)	12, 32	Average project length (*APL*)	$APL = \frac{\sum_{t=1}^{N} S_{nt}}{N}$
No. of resource types	1	Standard deviation of the project length (*SDPL*)	$SDPL = \sqrt{\frac{\sum_{t=1}^{N}(S_{nt} - APL)^2}{N-1}}$
Resource availability	10, 15, 20	Timely project completion probability (*TPCP*)	$TPCP = P(S_{nt} \leq duedate)$
Mean work content (μ_i)	$U[10,50]$	Stability cost (*SC*)	$SC = \frac{\sum_{t=1}^{N}\sum_{i=1}^{n} w_i \lvert S_{it} - s_i \rvert}{N}$
Standard deviation of the work content (σ_i)	$U[1,5]$	Instability weight for the dummy start activity	0
Work content	$N(\mu_i, \sigma_i)$	Instability weights for the non-dummy activities	$P(w_i = q) = (21 - 2q)\%$ $q \in \{1,2,\cdots,10\}$
Due date	$s_n^{min} \times 1.2$	Instability weight for the dummy end activity	0 (finishing the project early or at the due date)
			38 (finishing the project late)
Simulation runs (*N*)	1000	Baseline schedule	Optimal baseline schedule (adapted branch-and-bound)
			Good feasible baseline schedule (Tabu search)

Remark: S_{nt} denotes the starting (and thus the completion) time of the dummy end activity of the project in simulation run t; s_i represents the planned starting time of activity i; S_{it} denotes the realized starting time of activity i in simulation run t

6 Computational results

In this section, the computational results for both small instances (12 activities with an availability of 10) and larger instances (32 activities with an availability of 10) will be shown. The computational results with an availability of 15 and 20 are very similar and will not be presented in this section. However, the computational results for all resource availabilities will be analyzed in section 7.

6.1 Computational results for small instances with 12 activities

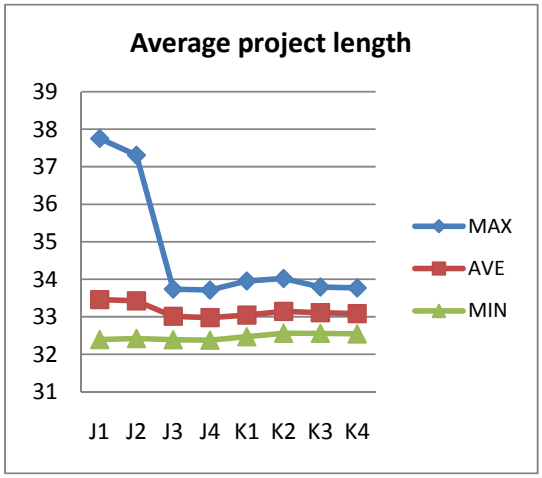

Fig. 5 *APL* of different priority lists and resource flow networks according to roadrunner scheduling and railway scheduling

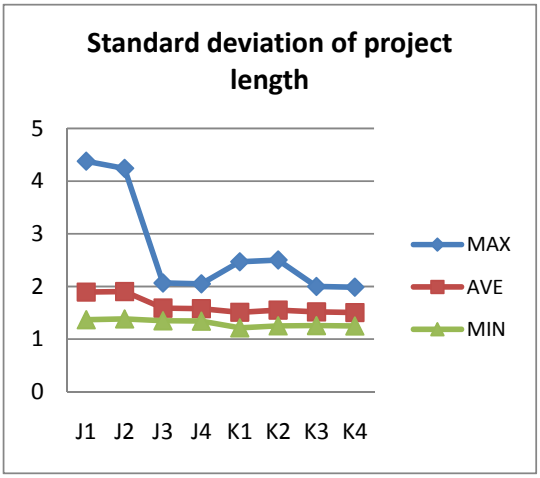

Fig. 6 *SDPL* of different priority lists and resource flow networks according to roadrunner scheduling and railway scheduling

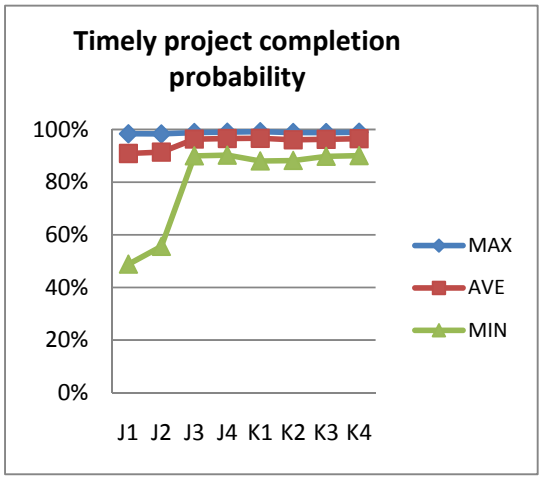

Fig. 7 *TPCP* of different priority lists and resource flow networks according to roadrunner scheduling and railway scheduling

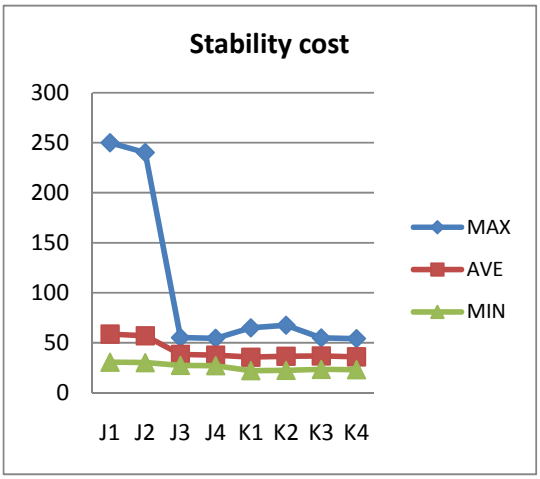

Fig. 8 *SC* of different priority lists and resource flow networks according to roadrunner scheduling and railway scheduling

In this section, we investigate the impact of the resource flow networks combined with the roadrunner scheduling policy and the railway scheduling policy for each schedule. Figures

5 to 8 represent the results for the two scheduling policies, where a 'J' indicates the roadrunner scheduling policy, a 'K' indicates the railway scheduling policy, number 1 represents execution according to the starting time priority list, number 2 represents execution according to the first critical chain priority list with a feeding buffer of 10%, number 3 represents execution based on Artigues et al.´s resource flow network and number 4 represents the execution based on the MABO resource flow network. For each resulting combination, the minimal, average and maximal result over all 100 instances is shown.

We can observe from Figures 5 to 8 that execution policies J1 and J2 perform worse than any other execution policy considering all four performance indicators. However, the average results of the other six execution policies are relatively close to each other, so it is really difficult to say which one is better than the other according to the figures. We decided to analyze the simulation results more in detail by using hypothesis tests for each sample population. This analysis can be found in section 7.

6.2 Computational results for larger instances with 32 activities

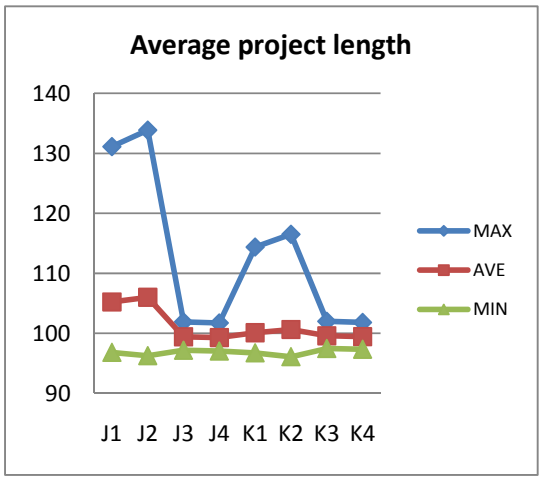

Fig. 9: *APL* of different priority lists and resource flow networks according to roadrunner scheduling and railway scheduling for larger instances

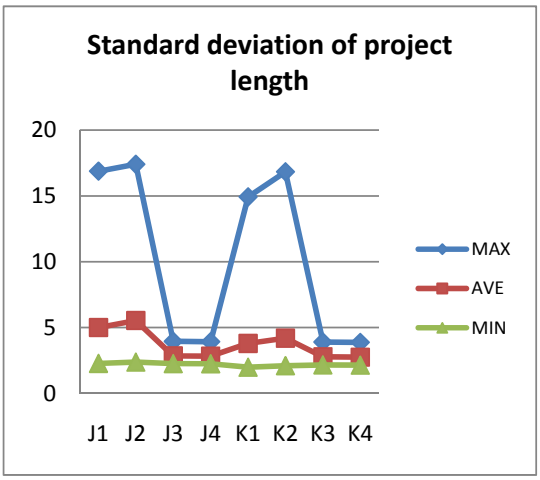

Fig. 10: *SDPL* of different priority lists and resource flow networks according to roadrunner scheduling and railway scheduling for larger instances

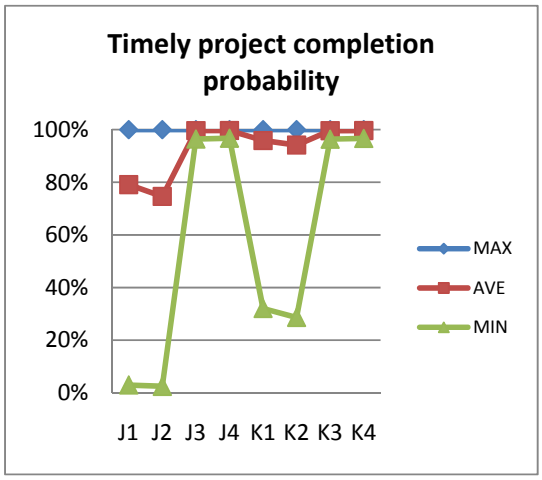

Fig. 11: *TPCP* of different priority lists and resource flow networks according to roadrunner scheduling and railway scheduling for larger instances

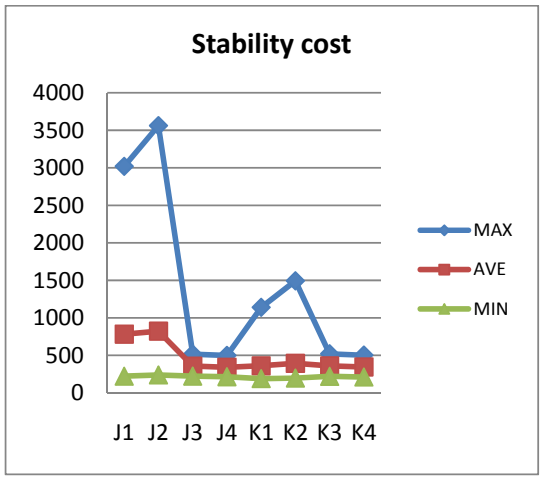

Fig. 12: *SC* of different priority lists and resource flow networks according to roadrunner scheduling and railway scheduling for larger instances

In this section, the computational results are shown for larger instances with 32 activities and an availability of 10. Figures 9 to 12 indicate that the behaviour of larger instances is quite similar to those of the small instances, and we can also conclude that execution policies J1 and J2 perform worse than any other execution policy considering all four performance indicators. The other six execution policies are not easy to compare as the average results are relative close to each other in the figures, although there is a hint that execution policies K1 and K2 perform slightly worse than the other four execution policies. Further analysis for these computational results can be found in section 7.

7 Analysis of the computational results by using SAS PROC GLM

In section 6, we have shown the computational results according to the eight execution policies for both small and somewhat larger problem instances. Unfortunately, it is much more difficult and complicated to compare the last six execution policies just from the figures. We decided to analyze our computational results by using SAS PROC GLM in this section. Section 7.1 SAS PROC GLM will be used to fit a multi-level mixed model for our computational results. The statistic output and interpretation will be shown in section 7.2 and the interaction between POLICY and APPROACH for both small instances and larger instances will be drawn in section 7.3.

7.1 A multi-level mixed model design

The data structure for the computational results is shown in Fig. 13. There are four levels which are indicated in capitals, namely POLICY, APPROACH, PR and ID. The POLICY level consists of two basic execution policies that are roadrunner scheduling and railway scheduling while the APPROACH level is composed of priority lists and resource flow networks. Those two levels are crossed over each other, resulting in four regions as follows:

- **Region A**: roadrunner scheduling + priority list (combination of roadrunner scheduling policy and priority list);
- **Region B**: roadrunner scheduling + resource flow networks algorithm (combination of roadrunner scheduling policy based on resource flow networks);
- **Region C**: railway scheduling + priority list (combination of railway scheduling policy and priority list);
- **Region D**: railway scheduling + resource flow networks algorithm (combination of railway scheduling policy based on resource flow networks).

The PR level is nested within the APPROACH level, which means that the priority list contains the starting time priority list (P_1) and the first critical chain priority list (P_2) while the resource flow networks contains Artigues et al.'s algorithm (R_1) as well as the MABO algorithm (R_2). The ID level contains 100 instances.

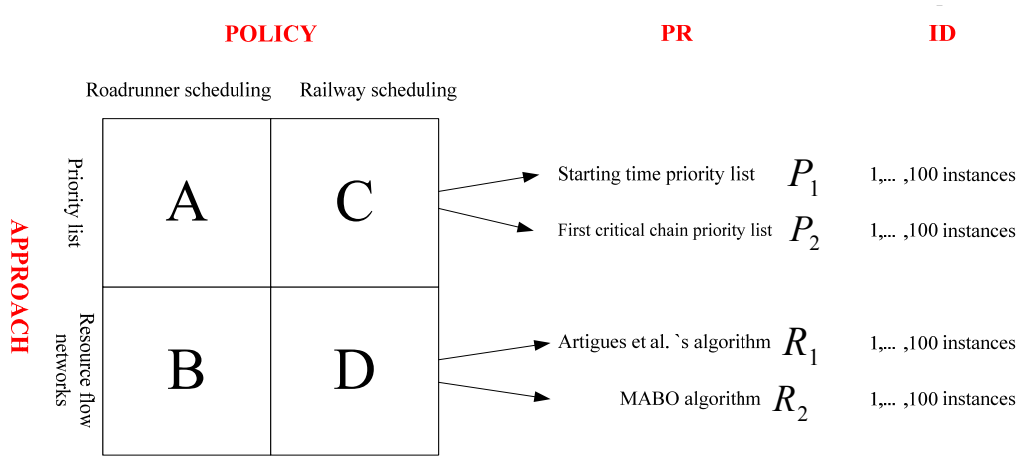

Fig. 13 Data structure for the computational results

The multi-level mixed model is $Y_{ijkl} = \mu + \alpha_i + \beta_j + (\alpha\beta)_{ij} + \tau_{k(ij)} + \varepsilon_{ijkl}$ $i = 1,2; j = 1,2; k = 1,2; l = 1,2,..,100$. Y_{ijkl} is the response variable which represents the performance indicator, μ is the overall mean and α_i, β_j and $(\alpha\beta)_{ij}$ are the fixed effects for POLICY, APPROACH and POLICY*APPROACH, respectively. $\tau_{k(ij)}$ is the fixed

effect of the k^{th} selected PR from the j^{th} APPROACH of the i^{th} POLICY, while ε_{ijkl} is the error term and is assumed iid $N(0, \sigma_{\varepsilon}^2)$.

The SAS PROC GLM statement for the APL performance indicators can be found in table 6. The SAS PROC GLM statements for the other performance indicators are similar to the statement for the APL except for the response variable in the model statement. Remark that we set a 0.05 significance level.

Table 6: SAS PROC GLM statement for the APL performance indicator

```
proc glm cl;
class POLICY APPROACH PR ID;
model APL= POLICY   APPROACH   POLICY*APPROACH   PR (APPROACH)   ID;
random ID;
lsmeans POLICY   APPROACH   POLICY*APPROACH   PR (APPROACH);
lsmeans POLICY   APPROACH   POLICY*APPROACH   PR (APPROACH)/pdiff;
run;
```

7.2 SAS output and interpretation

7.2.1 SAS output for the small instances test set with availability of 10

In this section, we conduct the SAS PROC GLM procedure to analyze the small instances test sets with an availability of 10. The SAS PROC GLM output for the APL performance indicator is shown in tables 7 to 9.

Table 7: Overall statistics information

The GLM Procedure					
	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	104	11599.57546	111.53438	586.96	<.0001
Error	695	132.06317	0.19002		
Corrected Total	799	11731.63863			

R-square	Coeff Var	Root MSE	APL Mean
0.988743	1.314748	0.435912	33.15554

Source	DF	Type I SS	Mean Square	F-Value	Pr > F
Policy	1	2.99017	2.99017	15.74	<.0001
Approach	1	9.82632	9.82632	51.71	<.0001
Policy*Approach	1	10.01600	10.01600	52.71	<.0001
PR(Approach)	2	0.09642	0.09642	0.51	0.6023
ID	99	11576.55012	116.93485	615.39	<.0001

Table 8: Least Square Means

Least Square Means										
Policy/Approach			Policy*Approach				PR(Approach)			
Policy	Approach	LSMEAN		Policy	Approach	LSMEAN		Approach	PR	LSMEAN
1		33.2167	A	1	1	33.4394	P_1	1	1	33.2513
2		33.0944	B	1	2	32.9940	P_2	1	2	33.2814
	1	33.2664	C	2	1	33.0933	R_1	2	1	33.0607
	2	33.0447	D	2	2	33.0955	R_2	2	2	33.0287

Table 9: Differences of Least Square Means

Least Squares Means for effect Policy*Approach Pr > |t| for H_0: LSMean(i)=LSMean(j)					Least Squares Means for effect PR(Approach) Pr >|t| for H_0: LSMean(i)=LSMean(j)				
i / j	A	B	C	D	*i / j*	P_1	P_2	R_1	R_2
A		<.0001	<.0001	<.0001	P_1		0.4905	<.0001	<.0001
B	<.0001		0.0229	0.0202	P_2	0.4905		<.0001	<.0001
C	<.0001	0.0229		0.9611	R_1	<.0001	<.0001		0.4631
D	<.0001	0.0202	0.9611		R_2	<.0001	<.0001	0.4631	

From table 7, it is clearly indicated that SAS PROC GLM fits our multi-level mixed model quite well as the R-square value equals 0.988743 which is very high. The p-values for the fixed effects of POLICY, APPROACH and POLICY*APPROACH are all smaller than 0.0001, and thus we can reject the null hypotheses: $\mu_{roadrunner\ scheduling} = \mu_{railway\ scheduling}$, $\mu_{priority\ list} = \mu_{resource\ flow\ networks}$ and $\mu_A - \mu_B = \mu_C - \mu_D$. In other words, there are significant differences for the average project length (APL) between railway scheduling and roadrunner scheduling, between the priority list and resource flow networks based approach as well as there is an interaction effect between POLICY and APPROACH.

From the left pane of table 9, it is clear that there is a significant difference between regions A and B, A and C, A and D, B and C and B and D (p-values below 5%). The only exception is that there is no significant difference between C and D at a significance level of 5%. Together with table 8, we can conclude that $\mu_B < \mu_C = \mu_D < \mu_A$ ("=" represents no significant difference, "B<C" represents that the mean of region B is significantly smaller than the mean of region C), which will be represented in the form of *B<C=D<A* in the left pane of table 10 (performance indicator APL for 12 activities and an availability of 10). Please remark here that we started out with a two-sided hypothesis test, but the resulting probabilities are so small that the LSMEAN values in the middle pane of table 8 clearly indicate which policy or approach is the better one if there is a significant difference between the two.

The p-value for the fixed effect of PR (APPROACH) in table 7 is 0.6023. As this value is larger than 0.05, we fail to reject the null hypothesis: $\mu_{P_1} = \mu_{P_2}$ and $\mu_{R_1} = \mu_{R_2}$, which means that the different priority lists and the different resource flow networks algorithms don't have a significant impact on the obtained results. In this case with an availability of 10, the fixed effect of PR (APPROACH) is not significant. However, in some other cases (for other availabilities or for the larger problem instances) the fixed effect of PR (APPROACH) is significant. Combining the right pane of table 8 with the right pane of table 9, we can tell which PR is better: table 9 indicates that there is no significant difference between P_1 and P_2 or between R_1 and R_2 while all other differences between a

priority list based approach and a resource network flow based approach are clearly significant, while table 8 indicates that the resource flow based approach leads to shorter project lengths. This results in the following relation $R_2 = R_1 < P_1 = P_2$ (the sequence of the R- and P-values in this relation is in sorted order of the APL-values). This relation is represented in the right pane of table 10 (again performance indicator APL for 12 activities and an availability of 10).

7.2.2 Summary for all the SAS output

There are so many SAS outputs for all test sets with different levels of resource availability that we don't show them all one by one. But from all the SAS outputs, we can conclude that for both small instances and larger instances, there is a significant difference between railway scheduling and roadrunner scheduling. Besides, there is also a significant difference between a policy based on a priority list and a policy based on a resource flow networks approach, and we can conclude that a policy based on a resource flow networks approach always outperforms a policy that is based on a priority list, not only for the small instances with 12 activities but also for the larger instances with 32 activities. The SAS outputs for all test sets and all performance indicators on the interaction between POLICY and APPROACH and PR (APPROACH) are summarized in table 10. The conclusion on the interaction between POLICY and APPROACH for both small and larger instances will be made in section 7.3.

Table 10: Summary of the SAS output

Act	Performance indicator	Availability			Availability		
		10	15	20	10	15	20
12	APL	**B**<C=D<A	**B=D=C**<A	**B=D**=C<A (B<C)	$\mathbf{R_2=R_1}<P_1=P_2$	$\mathbf{R_2=R_1}<P_1=P_2$	$\mathbf{R_2=R_1}<P_1<P_2$
	SDPL	**D=C**<B<A	**D=B=C**<A	**D=B**=C<A (D<C)	$\mathbf{R_2=R_1}<P_1=P_2$	$\mathbf{R_2=R_1}<P_1=P_2$	$\mathbf{R_2=R_1}<P_1<P_2$
	TPCP	**B=C**=D>A	**C=B=D**>A	**C**>B=D>A	$\mathbf{R_2=R_1}>P_1=P_2$	$\mathbf{R_2=R_1}>P_1=P_2$	$\mathbf{R_2}>R_1>P_1>P_2$
	SC	**C=D=B**<A	**C=D=B**<A	**C=D=B**<A	$\mathbf{R_2=R_1}<P_1=P_2$	$\mathbf{R_2=R_1}<P_1=P_2$	$\mathbf{R_2=R_1}<P_1<P_2$
32	APL	**B**<D<C<A	**B=D**<C<A	**B=D**<C<A	$\mathbf{R_2}<R_1<P_1<P_2$	$\mathbf{R_2}<R_1<P_1<P_2$	$\mathbf{R_2}<R_1<P_1<P_2$
	SDPL	**D=B**<C<A	**D=B**<C<A	**D=B**<C<A	$\mathbf{R_2=R_1}<P_1<P_2$	$\mathbf{R_2=R_1}<P_1<P_2$	$\mathbf{R_2}<R_1<P_1<P_2$
	TPCP	**B=D**>C>A	**D=B**>C>A	**D=B**>C>A	$\mathbf{R_2=R_1}>P_1>P_2$	$\mathbf{R_1=R_2}>P_1>P_2$	$\mathbf{R_2}>R_1>P_1>P_2$
	SC	**B=D**<C<A	**B=D**<C<A	**B=D**<C<A	$\mathbf{R_2}<R_1<P_1<P_2$	$\mathbf{R_2}<R_1<P_1<P_2$	$\mathbf{R_2}<R_1<P_1<P_2$

7.3 Conclusion on the interaction between POLICY and APPROACH

7.3.1 Conclusion for the small instances with 12 activities

From the summary of the SAS outputs for the small instances test sets (12 activities with an availability of 10, 15 and 20) in table 10, we can draw the conclusions for each performance indicator that are shown in the interaction matrix diagrams from Figures 14 to 17. In each interaction matrix diagram, we can see that there are four regions A, B, C and D and the mathematical symbols on a border of the matrix are meant to show the relationship between the two adjacent groups. Additionally, the symbol in the upper-left

corner of the A region indicates the relationship between regions A and D, while the symbol in the lower-left corner of the B region represents the relationship between regions B and C. Please remark that a 'greater than' symbol in these interaction matrix diagrams means significantly better (smaller for APL, SDPL and SC, larger for TPCP) or strictly dominating while a 'greater than or equal to' symbol means at least better or weakly dominating: a relationship is indicated as strictly dominating if the difference between the means of the two groups was significant for all three resource availabilities, it is indicated as weakly dominating if it is significantly different in only 1 or 2 out of the 3 availabilities (in that case, the value on the right side of the "greater (smaller) than or equal to" sign represents the probability of "greater (smaller) than"). As one could see in Fig. 14, region A is strictly dominated by regions B, C and D, and region B weakly dominates regions C and D with a probability of 1/3 and 2/3 respectively for the average project length. Additionally, there is no significant difference between C and D. From this figure, we can conclude that B dominates A, C and D with respect to the average project length (the dominant region for each performance indicator is indicated with a circle). We also can conclude that D dominates the other three regions for the standard deviation of the project length in Fig. 15, that C dominates the other three regions for the timely project completion probability in Fig. 16 and that A is dominated by B, C and D for the stability cost in Fig. 17. Remark that for none of the performance indicators the dominant group dominates the three other groups strongly.

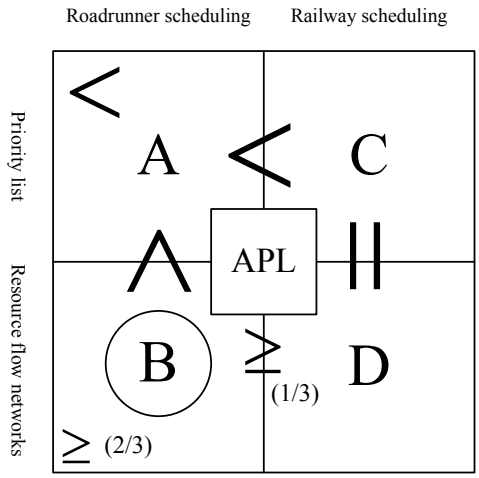

Fig. 14: Interaction matrix diagram for APL

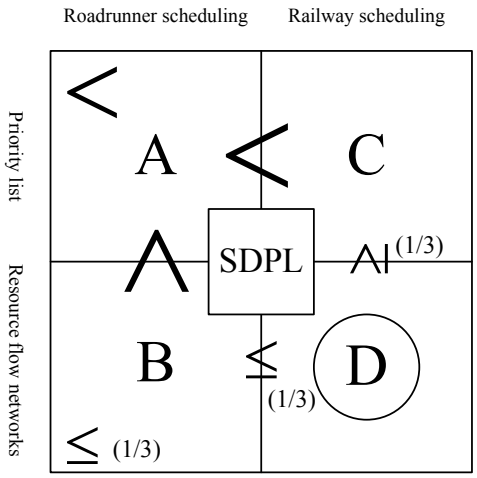

Fig. 15: Interaction matrix diagram for SDPL

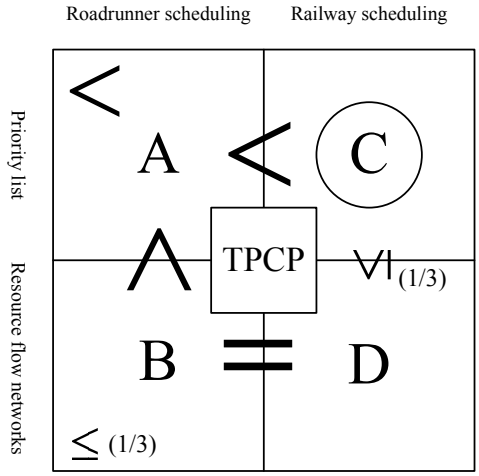

Fig. 16: Interaction matrix diagram for TPCP

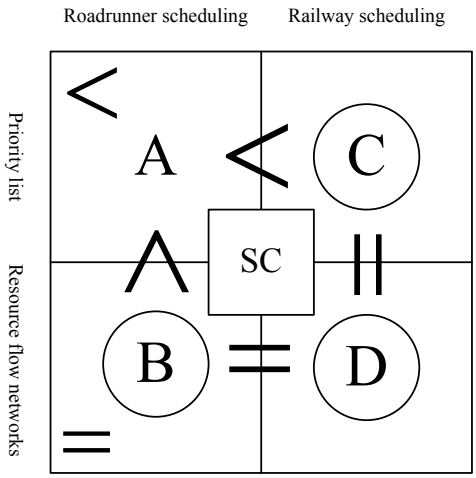

Fig. 17: Interaction matrix diagram for SC

7.3.2 Conclusion for larger instances with 32 activities

According to the summary of the SAS output for larger projects (32 activities with an availability of 10, 15 and 20), we can draw the conclusions for each performance indicator that are shown in the interaction matrix diagrams from Figures 18 to 21. The interaction matrix diagrams in Fig. 18 to 21 are different from the interaction matrix diagrams in Fig. 14 to 17, in the sense that all comparisons now show a strong relationship except B and D (weakly dominating relationship for APL, no significant difference between B and D for SDPL, TPCP and SC). We can conclude that B dominates the other three regions when considering the average project length, and that B and D strongly dominate the other two regions with respect to the standard deviation of the project length, the timely project completion probability and the stability cost.

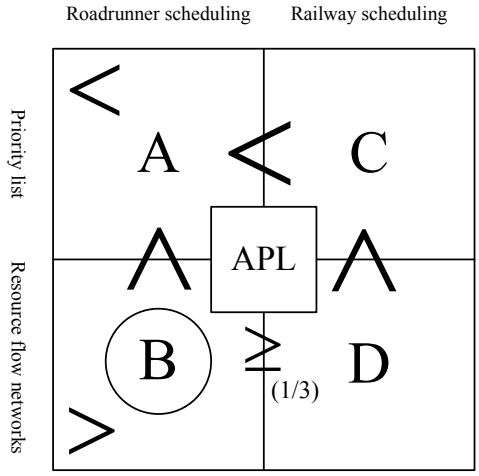

Fig. 18: Interaction matrix diagram for APL

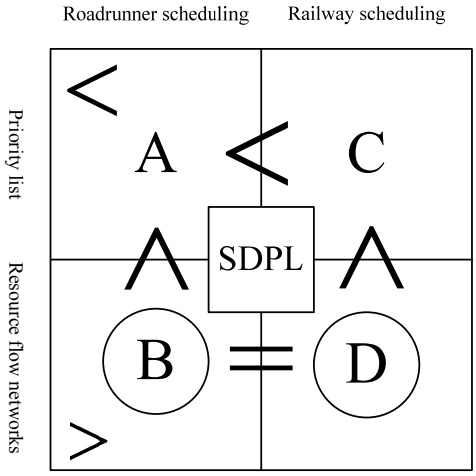

Fig. 19: Interaction matrix diagram for SDPL

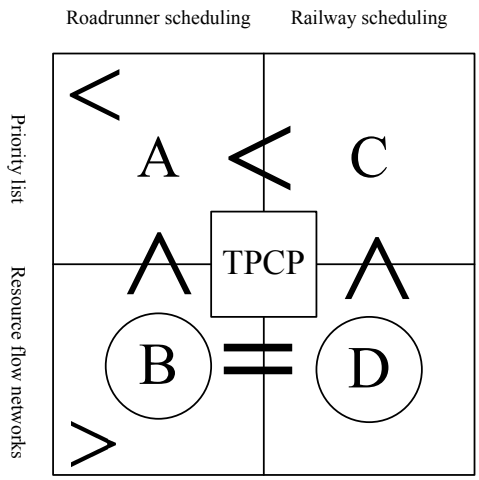

Fig. 20: Interaction matrix diagram for TPCP

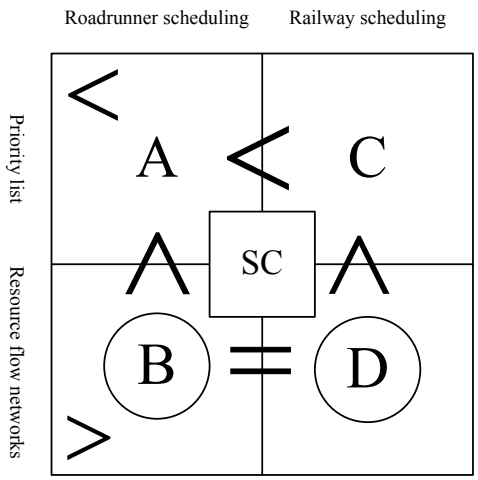

Fig. 21: Interaction matrix diagram for SC

8 Conclusions

In this paper, which continues the work of Tian & Demeulemeester (2010), we introduce the concept of resource flow networks and design a computational experiment to investigate the interaction between roadrunner or railway scheduling and the resource flow network. In our computational experiment, we found that in this project environment the starting time and first critical chain priority lists combined with roadrunner scheduling result in the worst outcomes when considering all four performance indicators. But the average results of two resource flow network based algorithms in combination with roadrunner scheduling and railway scheduling, as well as the starting time and the first critical chain priority list combined with railway scheduling seemed close to each other. As the differences between the average results didn't show very clearly in the experiment, we conducted a SAS PROC GLM procedure to fit a multi-level model and analyzed the interaction among the combinations of scheduling policies in detail. The statistical analysis results indicate that in a small project environment the choice of the combination strongly depends on the performance indicator of interest: for APL one prefers roadrunner scheduling based on a resource flow network, for SDPL the combination of railway scheduling based on a resource flow network seems better, for TPCP railway scheduling should be combined with a priority list, while for SC only evidence that roadrunner scheduling based on a priority list should not be used. Based on the analysis for the 32-activity networks, the combination of roadrunner scheduling based on a resource flow network seems optimal. Moreover, the SAS output also indicated that the railway scheduling seems better than the roadrunner scheduling, mainly because it is more robust with regard to the choice between a priority list or a resource flow network based

approach.

As we can see that the resource flow networks were generated based on a baseline schedule, the computational results depend heavily on the baseline schedule. Furthermore, there are many different algorithms for obtaining the resource flow networks and computational results based on different algorithms can have different effects on the final results. Therefore, how to generate stable baseline schedules and different applications of resource flow network algorithms can be topics for further research.

In this paper, we have enumerated all mode combinations in order to obtain those that result in a minimal project length. However, for the somewhat larger projects we have used some good feasible baseline schedules. Determining how to select the mode combinations which have the best performance when one is unable to enumerate all possible mode combinations could be another interesting topic for future research.

Acknowledgements

This research has been supported by the China Scholarship Council. We would like to acknowledge the China Scholarship Council for the financial support and the Research Center for Operations Management of the Katholieke Universiteit Leuven for providing a visiting research period to Wendi Tian. We would also like to acknowledge Bert De Reyck and Filip Deblaere for providing code that was indispensable for this research. Additionally, we would like to acknowledge Professor Martina Vandebroek for the help with the statistical analysis.

References

Artigues, C., Michelon, P., Reusser, S. (2003). Insertion techniques for static and dynamic resource-constrained project scheduling. European Journal of Operational Research, 149(2), 249–267.

Brucker, P., Drexl, A., Möhring, R., Neumann, K., Pesch, E. (1999), Resource-constrained project scheduling: Notation, classification, models, and methods, European Journal of Operational Research, 112, 3-41.

De, P., Dunne, E.J., Ghosh, J.B., Wells, C.E. (1997). Complexity of the discrete time-cost tradeoff problem for project networks. Operations Research, 45(2), 302-306.

De Reyck, B. (1998). Scheduling projects with generalized precedence relations: Exact and heuristic procedures, Ph.D. Thesis, Department of Applied Economics, Katholieke Universiteit Leuven, Leuven, Belgium.

De Reyck, B., Demeulemeester, E., Herroelen, W. (1998). Local search methods for the discrete time/resource trade-off problem in project networks, Naval Research Logistics, 45(6), 553-578.

Deblaere, F., Demeulemeester, E., Herroelen, W., Van de Vonder, S. (2007). Robust resource allocation decisions in resource-constrained projects. Decision sciences, 38(1), 5 - 37.

Demeulemeester, E., De Reyck, B., Herroelen, W. (2000). The discrete time/resource trade-off problem in project networks: A branch and bound approach, IIE Transactions, 32(11), 1059–1069.

Demeulemeester, E. & Herroelen, W. (2002). Project Scheduling: A research handbook. Kluwer Academic Publishers.

Goldratt, E.M. (1997). Critical Chain. New York: North River Press.

Hartmann, S. Briskorn, D. (2010), A survey of variants and extensions of the resource-constrained project scheduling problem, European Journal of Operational Research, 207, 1-14.

Herroelen, W., Demeulemeester, E., De Reyck, B. (1999). A classification scheme for project scheduling. In Jan Weglarz (Ed.), Project scheduling: Recent models, algorithm, and applications (pp. 1–26). Kluwer Academic Publishers.

Herroelen, W., Demeulemeester, E., De Reyck, B. (2000). A note on the paper "Resource-constrained project scheduling: Notation, classification, models, and methods" by Brucker et al., European Journal of Operational Research, 128(3), 679-688.

Kolisch, R., Sprecher, A. (1997). PSPLIB – A project scheduling problem library. European Journal of Operational Research, 96(1), 205–216.

Leus, R. (2003). The generation of stable project plans. PhD Thesis, Department of Applied Economics, Belgium: Katholieke Universiteit Leuven.

Leus, R., Herroelen, W. (2004). Stability and resource allocation in project planning. IIE Transactions, 36(7), 667-682.

Long, L., Ohsato, A. (2008). Fuzzy critical chain method for project scheduling under resource constraints and uncertainty. International Journal of Project Management, 26, 688-698.

Policella, N., Oddi, A., Smith, S., Cesta, A. (2004). Generating robust partial order schedules. 10th International Conference on the Principles and Practice of Constraint Programming – CP 2004 Proceedings, Toronto, Canada: Springer, 3258, 496–511.

Policella, N. (2005). Scheduling with uncertainty – A proactive approach using partial order schedules. PhD Thesis, Italy: Universita degli Studi di Roma "La Sapienza."

Ranjbar, M., Kianfar, F. (2007) Solving the discrete time/resource trade-off problem in project scheduling with genetic algorithms. Applied Mathematics and Computation, 191(2), 451-456.

Ranjbar, M., De Reyck, B., Kianfar, F. (2009). A hybrid scatter search for discrete time/resource trade-off problem in project scheduling. European Journal of Operational Research, 193(1), 35-48.

Tian, W., Demeulemeester, E. (2010) Railway scheduling reduces the expected project makespan. FBE Research Report KBI_1004, pp. 1 - 32.

Van de Vonder, S., Demeulemeester, E., Herroelen, W. (2008). Proactive heuristic procedures for robust project scheduling: An experiment analysis. European Journal of Operational Research, 189(3), 723-733.

Van Peteghem, V., Vanhoucke, M. (2010). A genetic algorithm for the preemptive and non-preemptive multi-mode resource-constrained project scheduling problem. European Journal of Operational Research, 201(2), 409-418.